WHAT IS CLAIMED IS:

1. A floating point execution unit for performing multiply/add operations using a plurality of operands taken from an instruction having a plurality of operand positions, the floating point unit comprising:

a multiplier for calculating a product of two of the operands;

an aligner coupled to the multiplier for combing said product and a third of the operands;

a first data path for supplying to the multiplier operands from a first and a second of the operand positions of the instruction;

a second data path for supplying the third operand to the aligner; and

a multiplexer on the second data path for selecting, for use by the aligner, either the operand from the second operand position of the instruction or the operand from the third operand position of the instruction.

2. A floating point execution unit according to Claim 1, wherein the first data path is maintained free of multiplexer operations.

3. A floating point execution unit according to Claim 1, wherein:

the aligner includes means to compute a shift amount for aligning said product and the third operand; and

the multiplexer operates to select the third operand in parallel with the means to compute the shift amount.

4. A floating point execution unit according to Claim 3, wherein the multiplexer selects the third operand while the means to compute computes said shift amount.

5. A floating point execution unit according to Claim 3, wherein each of the operands and said product includes an exponent value, and the means to compute computes said shift amount based only on said exponent values.

6. A floating point execution unit according to Claim 1, wherein each of the operands has an exponent value, and further comprising means, operating in parallel with the multiplier and the aligner, to determine whether the exponent values of any of the operands is zero.

7. A floating point execution unit according to Claim 6, wherein said means to determine tests said exponent values for a zero value while the multiplier calculates said product.

8. A floating point execution unit according to Claim 6, wherein the means to determine establishes a test result number based on results of said determination.

9. A floating point execution unit according to Claim 8, wherein:

the test result number includes a plurality of bits;

a first of the bits indicates whether the addend is zero; and

a second of the bits indicates whether the product is zero.

10. A floating point execution unit according to Claim 9, wherein the plurality of bits are used to force special values into the aligner result.

11. A floating point execution unit according to Claim 3, wherein the means to compute the shift amount compresses two of the three input exponents and an offset while selecting the third exponent.

12. A floating point execution unit according to Claim 11, wherein, when executing an add or subtract instruction, the means to compute the shift amount computes the alignment shift amount as ea + eb – 2eb.

13. A method of operating a floating point execution unit to perform multiply/add operations the floating point unit having a multiplier, an aligner coupled to the multiplier, and a multiplexer, the method comprising the steps:

sending an instruction to the floating point unit, the instruction having a plurality of operand positions holding operands;

using the multiplier to calculate a product of two of the operands;

using the aligner to combine said product and a third of the operands;
supplying over a first data path to the multiplier operands from a first and a second of the operand positions of the instruction;

supplying over a second data path the third operand to the aligner; and

positioning the multiplexer on the second data path;

using the multiplexer to select, for use by the aligner, either the operand from the second operand position of the instruction or the operand from the third operand position of the instruction.

14. A method according to Claim 13, comprising the further step of maintaining the first data path free of multiplexer operations.

15. A method according to Claim 13, comprising the further step of:

using the aligner to compute a shift amount for aligning said product and the third operand; and wherein

the multiplexer operates to select the third operand in parallel with the aligner.

16. A method according to Claim 15, wherein the multiplexer selects the third operand while the aligner computes said shift amount.

17. A method according to Claim 15, wherein each of the operands and said product includes an exponent value, and the step of using the aligner to compute said shift amount includes the step of computing said shift amount based only on said exponent values.

18. A method according to Claim 13, wherein each of the operands has an exponent value, and comprising the further step of, determining, in parallel with the multiplier and the aligner, whether the exponent values of any of the operands is zero.

19. A method according to Claim 18, wherein the step of determining whether the exponent values of any of the operands is zero occurs while the multiplier calculates said product.

20. A method according to Claim 18, comprising the further steps of:

establishing a test result number based on results of said determination, the test result number including a plurality of bits,

using a first of the bits to indicate whether the addend is zero; and

using a second of the bits to indicate whether the product is zero.